まい USB 取扱説明書（HighSpeed 対応版）

株式会社 北斗電子

（REV.1.0.4.0）

目　　次

1. はじめに

ちょっとした周辺機器をつなごうとするとき、シリアル・ポートなどのインターフェースを使用することが一般的でした。

最近のパソコンでは、シリアル・ポートが存在しないことも少なくなく、USB ポートを利用することが主流になってきました。ターゲット機器ではシリアル・ポートがお手軽であるため、USB シリアル変換を使っています。

しかし、USB シリアル変換を使用するとユーザに仮想 COM ポートを選択させることが必要となる上、通信速度の低いものになってしまいます。

USB にはシリアルよりはるかに速い伝送速度を実現できるといったメリットがあるばかりか、USB 電源を活用することによりターゲットの電源周りをシンプルにできるメリットがあります。

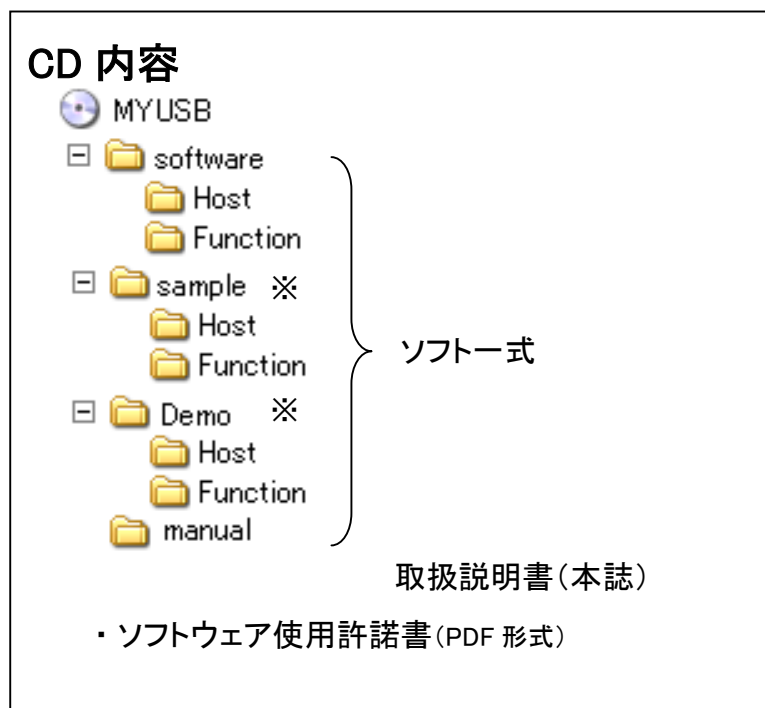
そういったメリットから、ターゲット機器を USB 化したいと思われる方々が、多くなってきています。しかし、USB 機器を開発しようとすると、具体的な作成事例などがあまり存在しないため、USB 規格、Windows ドライバ及びターゲット基板について十分に理解するのに多くの時間を要します。

「まい USB」はそういった難しい知識をアプリケーションに意識させることなく、まるでシリアル通信かのように USB 通信を実現するお手軽なライブラリソフトウェアです。
この取扱説明書はソフトウェアの使用方法について説明しています。
正しくご使用していただくために、この取扱説明書をよくお読みください。

2. 製品内容

次のものが全てそろっているかどうかをご確認ください。

- ＣＤ.........................1 枚
- ソフトウェア使用許諾書.......1 枚

※「myusb_XXXX.拡張子」の XXXX は製品 ID を意味しています。

※製品によっては、どちらか一方のご用意となります。

3. 概要

「まいUSB」は、ホスト（Windows[1]）とファンクション（ターゲット基板）で行うUSB通信を自動的に行うUSBハイスピード対応のソフトウェアライブラリです。

「まいUSB」のAPI関数をコールするだけで、USB通信を簡単に行うことができます。
API関数の詳細については、8章のAPI関数（ユーザーインターフェイス）をご参照ください。

「まいUSB」を使用したUSB通信の動作概念図を図 1に示します。

ホスト
ファンクション
アプリケーション
まい USB（ホスト）
まい USB（ファンクション）
アプリケーション
初期化関数コール
認証
初期化関数コール
データ送信関数コール
バルクアウト転送
データ受信関数コール
ホストからのデータを取得
データ送信関数コール
ファンクションからのデータを取得
データ受信関数コール
バルクイン転送

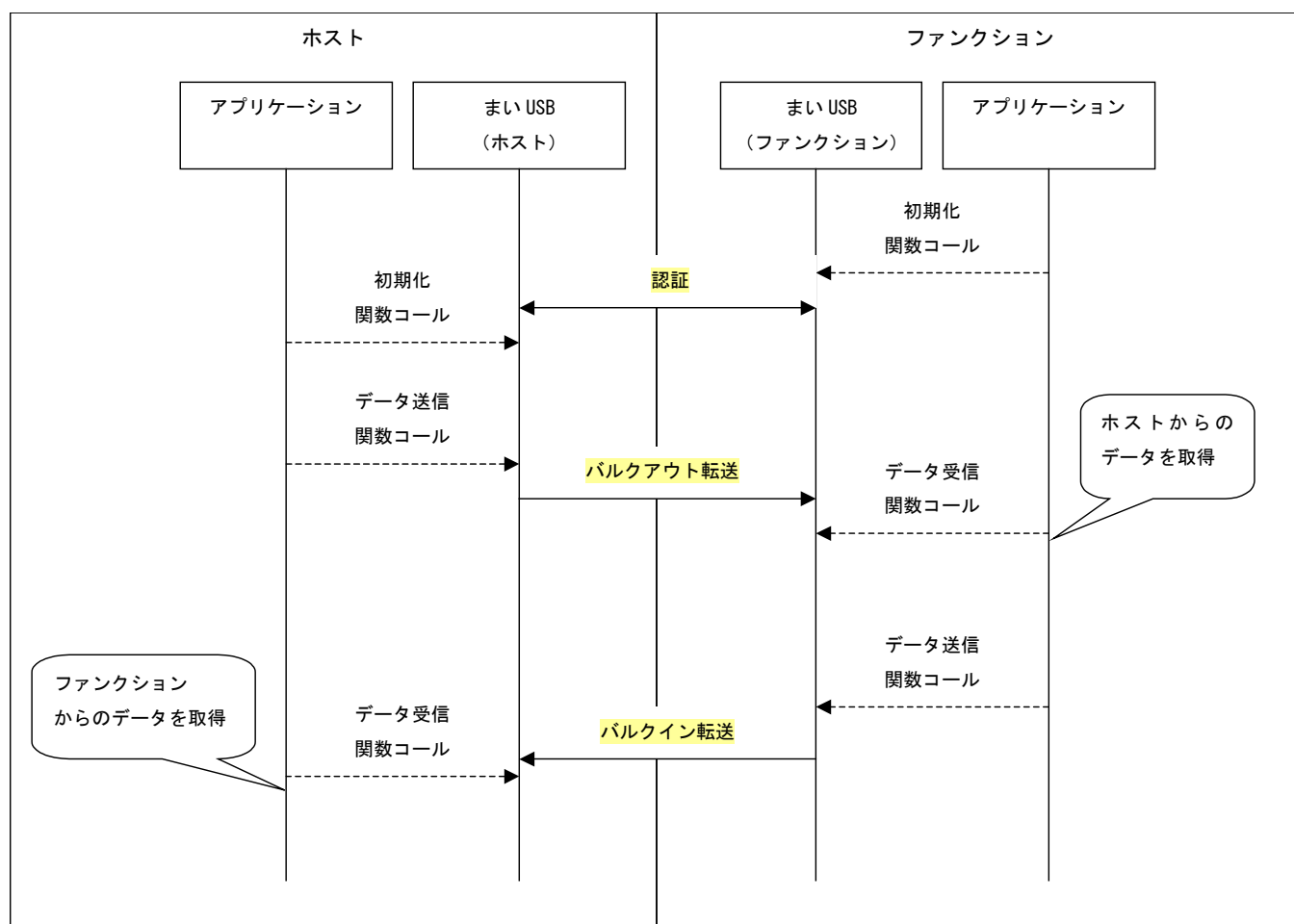



図 1「まい USB」動作概念図

[1] WindowsOS は、Windows2000, WindowsXP, WindowsVista, Windows7, Windows8.1 にて動作いたします。

4. 製品仕様

4.1. USB ホスト（Windows 側）

対応 OS	:	Windows2000 WindowsXP WindowsVista Windows7 Windows8.1
推奨 CPU スペック	:	3.00 GHz
推奨メモリサイズ	:	512 MByte
必要ディスクサイズ	:	30 KByte 以上

4.2. USB ファンクション（ターゲット基板側）

「まい USB」では以下の領域名で設定しており、以下のサイズを必要とします。
ご使用の際には、下記領域名を設定してください。

セクション名	領域名	サイズ
プログラム領域	PMYUSB	11KByte
定数領域	CMYUSB	256Byte
未初期化データ領域	BMYUSB	67Kbyte（受信データバッファ：65Kbyte）
スタック	－	64Byte

5. 「まいUSB」を組み込む方法（開発編）

5.1. USBホスト（Windows側）

・ API 関数をご使用になられるアプリケーションファイルに、「usb_host_dll.h」をインクルードするように実装してください。
・ 「myusb_XXXX.dll」をロードし、8.1.2章に記載していますAPI関数のポインタを取得するように実装してくさい。
・ アプリケーションの仕様に合わせて上記で取得した関数ポインタをご使用ください。
・ ビルドを行う際には、CDに入っています「usb_host_dll.h」を開発環境（インクルードパス上）にコピーしてください。

5.2. USBファンクション（ターゲット基板側）

・ API 関数をご使用になられるアプリケーションのファイルに、「usb_function_api.h」をインクルードするように実装してください。
・ アプリケーションの仕様に合わせて8.2.2章に記載していますAPI関数をご使用ください。
・ コンパイルを行う際には、CDに入っています「usb_function_api.h」を開発環境（インクルードパス上）にコピーしてください。
・ リンクを行う際には、CDに入っています「myusb_XXXX.lib」を開発環境（入力ライブラリファイル指定パス上）にコピーしてください。
　お使いのコンパイル/リンク環境によって以下の形式のものをご利用下さい。
　・バンク切換なしのELF形式: myusb_XXXX_elf.lib
　・バンク切換ありのELF形式: myusb_XXXX_resbank.lib(バンク切換に対応したマイコンのみ)

6. 「まいUSB」を組み込む方法（運用編）

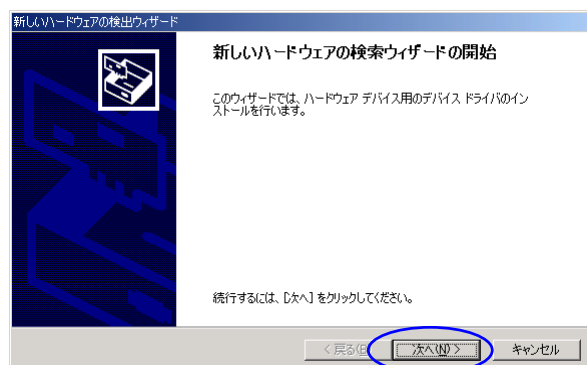
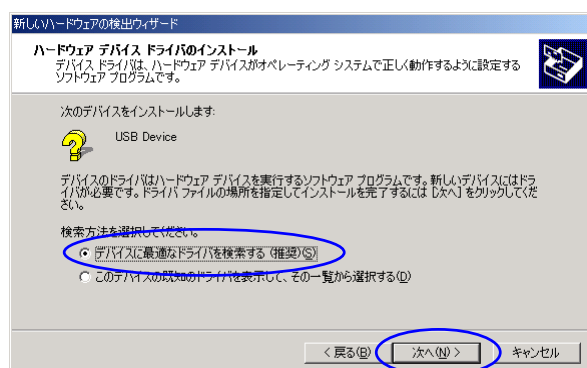
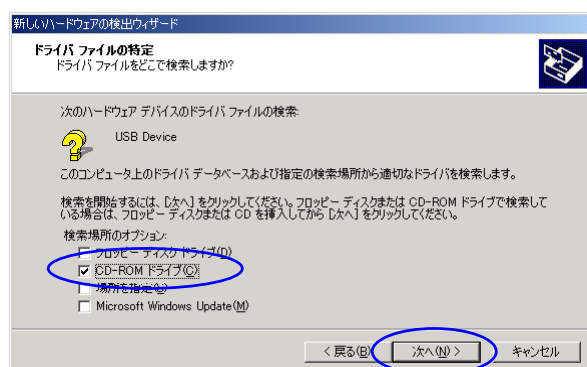
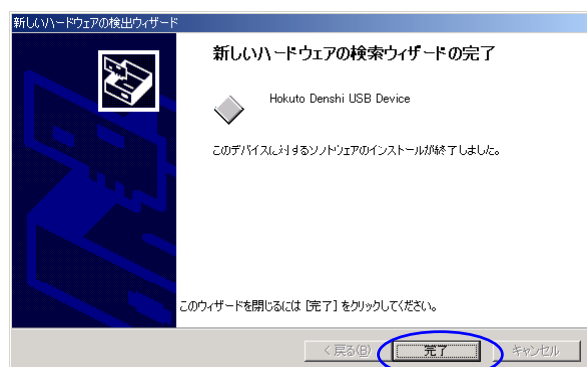
(1) 5.2章にて開発しましたUSBファンクションソフトウェアをお手持ちのWriterでターゲット基板に書き込んでください。
(2) ホストマシンの電源を投入し、システムを起動してください。
(3) ホストマシンとターゲット基板をUSBケーブルで接続し、ターゲット基板の電源を投入してください。
　※初回接続時には、ホストマシンにてドライバのインストールが促されますので、
　7章のWindowsドライバインストール方法をご参照の上、ドライバをイントールしてください。
(4) 5.1章にて開発しましたUSBホストアプリケーションとCDに入っています「myusb_XXXX.dll」を同じフォルダに格納してください。
(5) USBホストアプリケーションを起動してください。
(6) 完了

7. Windows ドライバインストール方法

■Windows2000 の場合

（1） ホストマシンに CD をドライブに入れてください。

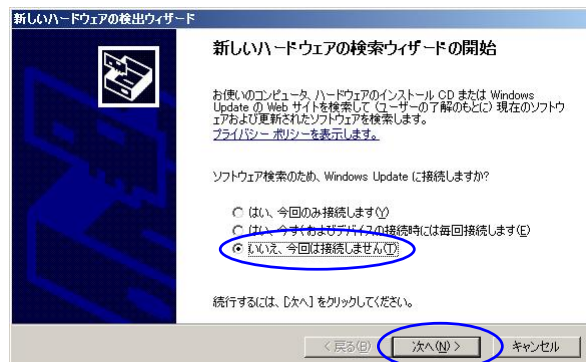
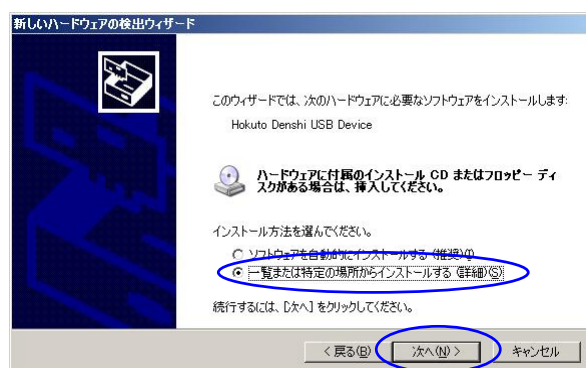
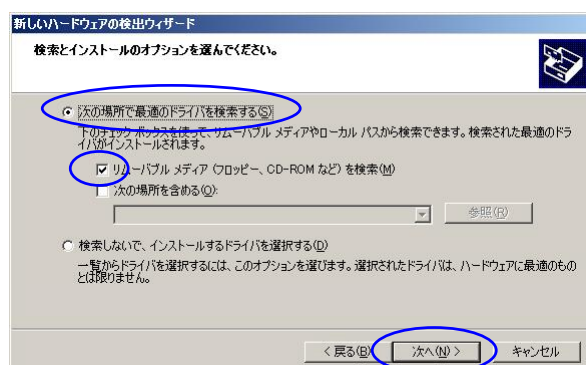
（2） CD が読み込まれたことを確認してからターゲット基板とホストマシンを USB にて接続してください。

（3） 自動的に右図のような画面が表示されますので、「次へ（N）」をクリックしてください。

（4） 「**デバイスに最適なドライバを検索する（推奨）（S）**」を選択し、「**次へ（N）**」をクリックしてください。

（5） 「**CD-ROM ドライブ（C）**」を選択し、「**次へ（N）**」をクリックしてください。

（6） 「**完了**」をクリックしてください。

■WindowsXP の場合

(1) ホストマシンに CD をドライブに入れてください。

(2) CD が読み込まれたことを確認してからターゲット基板とホストマシンを USB にて接続してください。

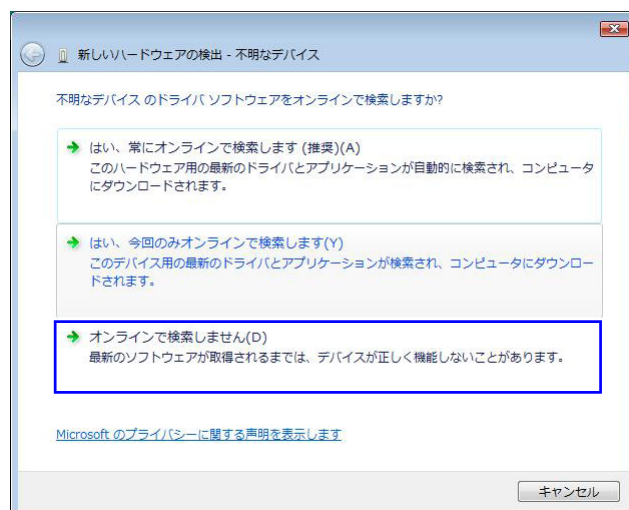
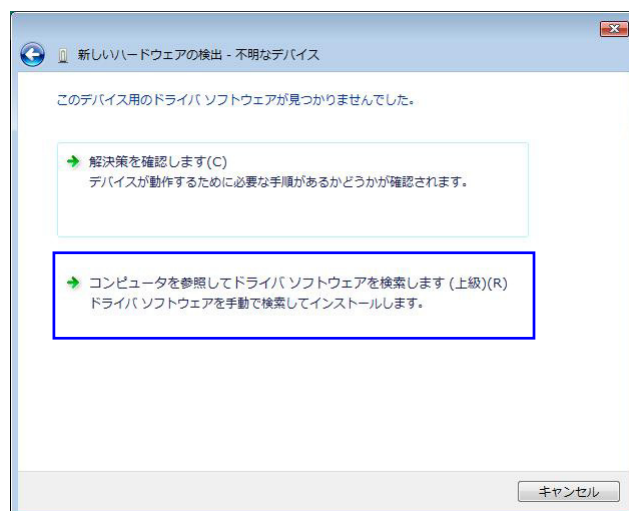
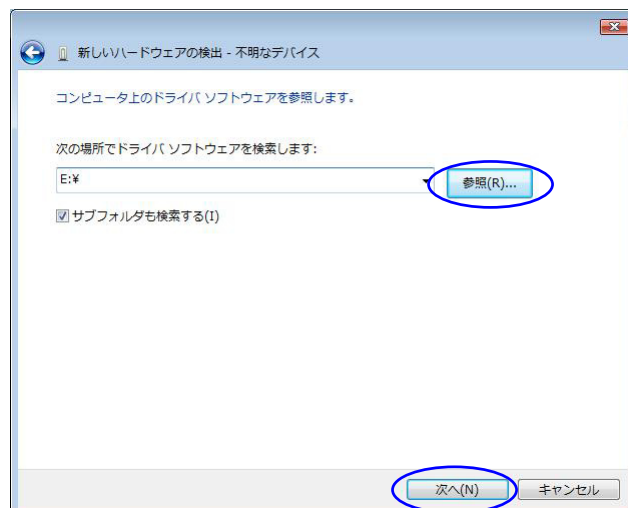
(3) 自動的に右図のような画面が表示されますので、「いいえ、今回は接続しません(T)」を選択し、「次へ（N）」をクリックしてください。

(4) 「一覧または特定の場所からインストールする（詳細）（S）」を選択し、「次へ（N）」をクリックしてください。

(5) 「次の場所で最適のドライバを検索する（S）」と「リムーバブルメディア（フロッピー、CD－ROMなど）を検索（M）」を選択し、「次へ（N）」をクリックしてください。

(6) 「完了」をクリックしてください。

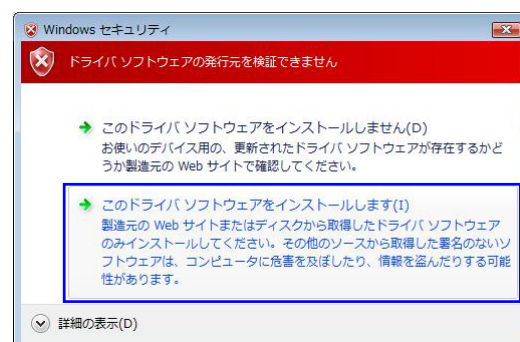
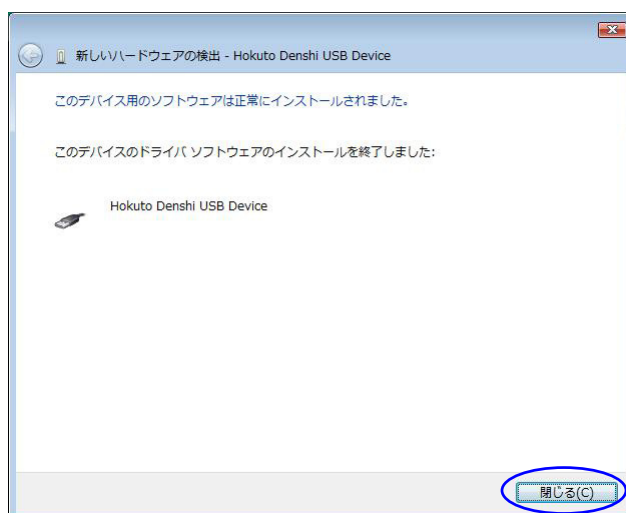
■WindowsVista の場合

(1) ホストマシンに CD をドライブに入れてください。

(2) CD が読み込まれたことを確認してからターゲット基板とホストマシンを USB にて接続してください。

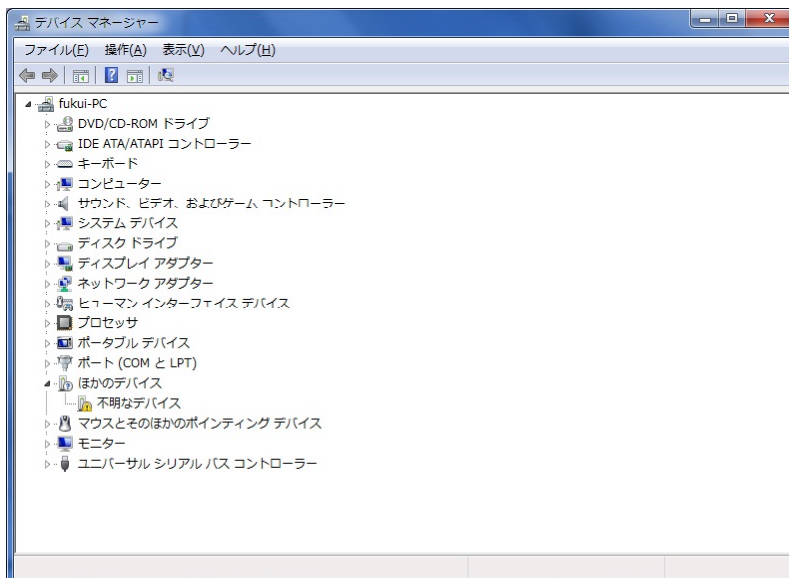
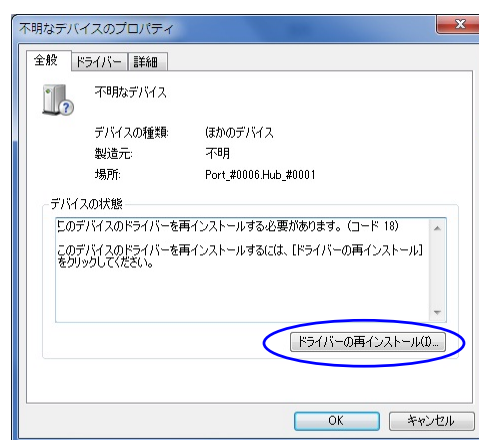
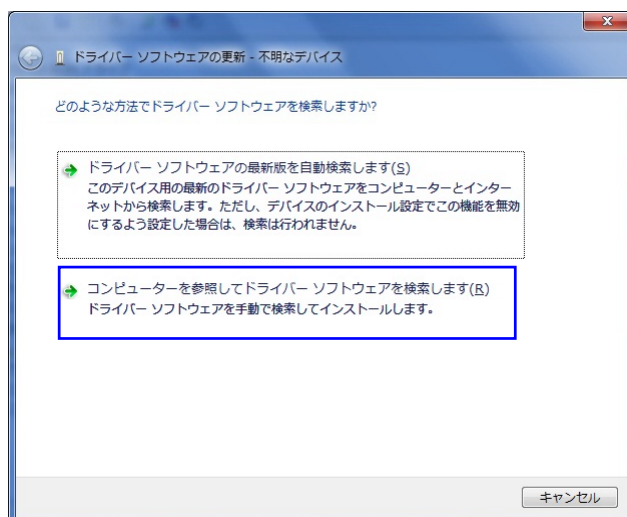
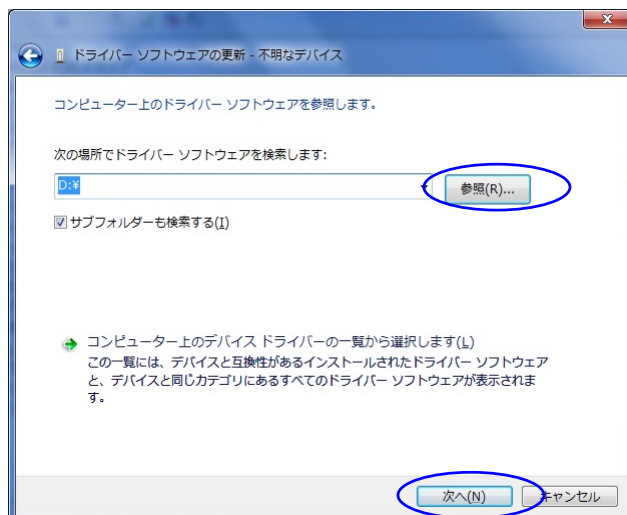
(3) 自動的に右図のような画面が出力されますので、「**ドライバソフトウェアを検索してインストールします（推奨）**」をクリックしてください。

(4) 「**オンラインで検索しません（D）**」をクリックしてください。

(5) 「**コンピュータを参照してドライバソフトウェアを検索します（上級）（R）**」をクリックしてください。

(6) 「**参照**」をクリックし、お使いの CD ドライブを選択し、「**次へ（N）**」をクリックしてください。

(7) 「**このドライバソフトウェアをインストールします（I）**」をクリックしてください。

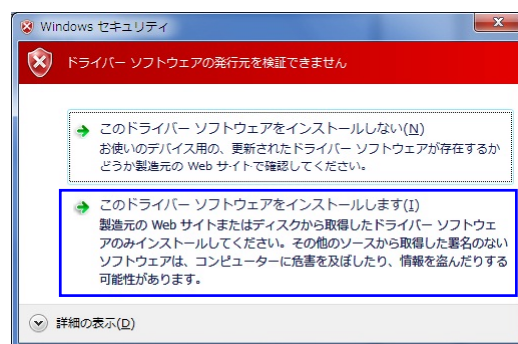
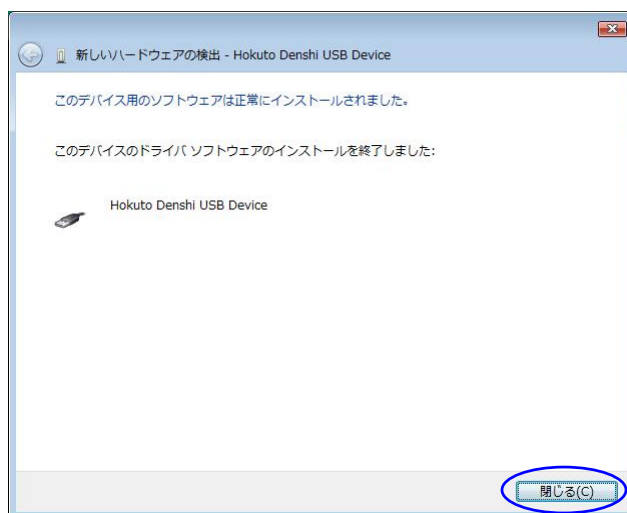
右図のような警告が表示されますが、弊社で十分な検証を行っていますので安心してご使用ください。

(8) 「**閉じる（C）**」をクリックしてください。

■Windows7/8.1 の場合

(1) ホストマシンに CD をドライブに入れてください。

(2) CD が読み込まれたことを確認してからターゲット基板とホストマシンを USB にて接続してください。

(3) タスクバーに「デバイスドライバは正しくインストールされませんでした」が表示されますので、「デバイスマネージャー」画面を開いてください。

(4) 不明なデバイスをダブルクリックすると右図のような画面が出力されますので、**「ドライバーの再インストール」**をクリックしてください。 ※Windoes8.1 の場合**「ドライバーの更新」**と表示されます。

(5) **「コンピューターを参照してドライバーソフトウェアを検索します(R)」**をクリックしてください。

(6) 「**参照**」をクリックし、お使いの CD ドライブを選択し、「**次へ（N）**」をクリックしてください。

(7) 「**このドライバソフトウェアをインストールします（I）**」をクリックしてください。

右図のような警告が表示されますが、弊社で十分な検証を行っていますので安心してご使用ください。

※購入時期によっては本警告が出ない場合があります。

(8) 「**閉じる（C）**」をクリックしてください。

8. API 関数（ユーザーインターフェイス）

8.1. USB ホスト（Windows 側）

USB ホストの API 関数の実体は dll に実装されています。
アプリケーションから関数をコールする場合、dll をロードなどしてご使用ください。

8.1.1. API 関数一覧

USB ホストの API 関数一覧を表 8-1に示します。

表 8-1　USB ホスト API 関数一覧

関数名	説明
MyUsbHostInitDll	初期化関数
MyUsbHostTermDll	終了関数
MyUsbHostSendData	データ送信関数
MyUsbHostRecvData	データ受信関数
MyUsbHostGetLastEvent	最終イベント情報取得関数
MyUsbHostGetRecvPointer	受信バッファポインタ取得関数
MyUsbHostGetLastError	最終エラー状態取得関数

8.1.2.API 関数詳細

MyUsbHostInitDll

初期化関数

書式

```
unsigned char* MyUsbHostInitDll (HWND hWnd,
                                 unsigned long mssageStsNum,
                                 HANDLE hStsEvent,
                                 BOOL bSendBlock)
```

引数

引数		説明
HWND hWnd	I	イベントメッセージを受取るWindow ハンドル
unsigned long mssageStsNum	I	状態に変化があったとき（USB 切断時及び、データ送信完了時）に送信するメッセージ ID
HANDLE hStsEvent	I	状態に変化があったとき（USB 切断時及び、データ送信完了時）に送信するイベントハンドル
BOOL bSendBlock	I	データ送信ブロック指定 TRUE：送信完了するまでデータ送信関数を終了しない FALSE：送信完了を待たずにデータ送信関数を終了する

戻り値

受信データバッファポインタ

但し、以下の場合には「NULL」が返ります。

- PIPE が開けなかった場合（USB 切断中又は PIPE がすでに開いている）
- 内部リソースを取得できなかった場合（リソース不足）

詳細

使用する全ての PIPE を開き、正常終了した場合に、受信データのバッファポインタを返します。異常終了した場合には、NULL を返します。
※本関数は、USB 接続時に呼び出してください。
本関数の正常終了後、USB 通信が可能となります。

●引数について

○USB 切断

以下の条件のとき、ステータスに変化があったことをイベントメッセージにてアプリケーションに通知させることができます。

- ✓ hWnd が「NULL 以外」の場合には、hWnd のメッセージキューに mssageStsNum をポストします。
- ✓ hStsEvent が「NULL 以外」の場合には、hStsEvent をシグナル状態にします。

○データ送信完了

以下の条件のとき、ファンクションへデータを送信完了したことをアプリケーションに通知させることできます。

- ✓ bSendBlock が「FALSE」かつ hWnd が「NULL 以外」の場合には、hWnd のメッセージキューに mssageStsNum をポストします。
- ✓ bSendBlock が「FALSE」かつ hStsEvent が「NULL 以外」の場合には、hStsEvent をシグナル状態にします。

※アプリケーションへの通知を行わないようにする場合には、bSendBlock を「TRUE」に設定してください。

●イベント発生について
イベントを発生させる場合、イベント発生を契機にアプリケーションにてMyUsbHostGetLastEvent関数をコールすることによってイベントの種類を知ることができます。

MyUsbHostTermDll

終了関数

書式	void MyUsbHostTermDll (void)
引数	なし
戻り値	なし
詳細	MyUsbHostInitDll で生成したリソースを開放し、全ての PIPE を閉じます。 アプリケーション終了時、及び USB 切断時（MyUsbHostGetLastEvent関数の戻り値が EVENT_DISCONNECT）には必ず本関数をコールしてください。

MyUsbHostSendData

データ送信関数

書式	BOOL MyUsbHostSendData(unsigned char* pData, unsigned long len)
引数	unsigned char* pData　I　送信データバッファポインタ unsigned long len　I　送信データサイズ(1～0x10000)
戻り値	TRUE　：正常終了 FALSE　：異常終了 ・ PIPE が開いていない。 ・ 送信データバッファが NULL ・ 送信データ長が 0 又は 0x10001 以上 ・ 接続が切れた場合 ・ データ送信中の場合 （MyUsbHostInitDll関数で bSendBlock を「FALSE」に設定している場合）
詳細	指定されたデータをデータサイズ分、ファンクションへ送信します。 ファンクション側ではMyUsbFunctionGetRecvData関数でデータを読み出してください。 一回のデータ送信サイズは 0x10000Byte までですので、0x10000Byte を超えるデータを送信する場合には、複数回コールしてください。 ホストはファンクションの受信 FIFO があふれていてもデータ送信を行うことができます。 ファンクションの受信 FIFO がオーバーフローを起こさないようにする場合、アプリケーションで制御を行ってください。

MyUsbHostRecvData

データ受信関数

書式	BOOL MyUsbHostRecvData(unsigned long* pLength)
引数	unsigned long* pLength　　O　受信データサイズポインタ 以下の場合には、「0」が返ります。 ・ PIPE が開いていない。 ・ 受信データがない場合 ・ 接続が切れた場合
戻り値	TRUE　：正常終了 FALSE　：異常終了 ・ PIPE が開いていない。 ・ 接続が切れた場合
詳細	ファンクションからMyUsbFunctionSetSendData関数で送信されたデータを受信し、受信データバッファ(MyUsbHostInitDll関数／MyUsbHostGetRecvPointer関数の戻り値）にデータを格納します。 アプリケーション側で受信データサイズ分、受信バッファからデータを読み出してください。 ホストで本関数を呼び出さないとファンクションの送信 FIFO があふれる可能性があります。 ファンクションの送信 FIFO がオーバーフローを起こさないようにする場合、アプリケーションで制御を行ってください。

MyUsbHostGetLastEvent

最終イベント情報取得関数

書式	unsigned long MyUsbHostGetLastEvent(void)		
引数	なし		
戻り値	EVENT_NON	(0)	：イベントなし（初期値）
	EVENT_FINISH_SEND	(1)	：送信完了イベント
	EVENT_DISCONNECT	(0xFFFFFFFF)	：切断イベント
詳細	最後に発生したイベント情報を取得することができます。		

MyUsbHostGetRecvPointer

受信バッファポインタ取得関数

書式	unsigned char* MyUsbHostGetRecvPointer(void)
引数	なし
戻り値	受信データバッファポインタ
詳細	初期化が正常終了している場合に、受信データのバッファポインタを返します。異常終了している場合、未初期化時又は、MyUsbHostTermDll関数をコールした場合には、NULL を返します。 本関数は、MyUsbHostInitDll関数で確保した受信バッファポインタを返すための関数であり、受信バッファポインタのデータを更新するものではありません。 データ更新するには、MyUsbHostRecvData関数をコールしてください。

MyUsbHostGetLastError

最終エラー状態取得関数

書式	unsigned long MyUsbHostGetLastError(void)
引数	なし
戻り値	エラー番号 0　　　　　　　：エラーなし 100～199　　　：初期化系エラー 200～299　　　：データ送信系エラー 300～399　　　：データ受信系エラー
詳細	DLL 内部で最後に発生したエラー状態を取得することができます。 ※本関数は USB 転送機能には一切影響ありません。アプリケーション作成時のデバッグ用としてご使用ください。

8.2. USB ファンクション（ターゲット基板側）

USB ファンクションはライブラリ形式となっています。
アプリケーションからは、ライブラリをリンクすることで、ライブラリの API 関数をコールすることができます。

最大 512Byte×129 個のバッファリングが可能な受信 FIFO をもっており、ホストからの送信データを「まい USB」にて受信 FIFO に格納し、MyUsbFunctionGetRecvData 関数でデータを読み出すことができます。
ホストへのデータ送信は、アプリケーションにて MyUsbFunctionSetSendData 関数コール時に設定したデータを「まい USB」にて送信します。

【注意事項】
- 割り込みベクタテーブルにUSB割り込み関数「usb_function_interrupt」をご登録ください。
- USBを制御するにあたりターゲット基板に一部制約がある場合がございます。
  詳細は、「9章ターゲット基板別制約」をご参照ください。

8.2.1. API 関数一覧

USB ファンクションの API 関数一覧を表 8-2、表 8-3に示します。

表 8-2　USB ファンクション API 関数一覧

関数名	説明
MyUsbFunctionInit	初期化関数
MyUsbFunctionIsConnection	USB デバイス状態取得関数
MyUsbFunctionSetSendData	データ送信関数
MyUsbFunctionGetSendStatus	データ送信状態取得関数
MyUsbFunctionGetRecvData	データ受信関数
MyUsbFunctionGetRecvDataLength	受信データ長取得関数
MyUsbFunctionGetRecvStatus	データ受信状態取得関数
MyUsbFunctionClearRecvErrorStatus	データ受信状態クリア関数
MyUsbFunctionGetRecvFifoCouter	受信 FIFO 数取得関数
MyUsbFunctionSetPriority	USB 割り込み優先度設定関数

表 8-3　USB ファンクション割り込み処理関数一覧

関数名	説明
usb_function_interrupt	USB ファンクション割り込み処理関数

8.2.2.API 関数詳細

以下の API 関数の戻り値として記載しています「BOOL」は「unsigned char」 を定義したものです。
TRUE は「1」、FALSE は「0」を示します。
尚、関数定義及び、定数定義については、「usb_function_api.h」に記載しています。

MyUsbFunctionInit
初期化関数

書式	void MyUsbFunctionInit(void)
引数	なし
戻り値	なし
詳細	USB ライブラリを初期化します。 ※本関数は、ターゲット基板起動時のみ呼び出してください。 　正しくホストと接続するために約 2 秒ほどの時間がかかります。

MyUsbFunctionIsConnection
USB デバイス状態取得関数

書式	BOOL MyUsbFunctionIsConnection(void)
引数	なし
戻り値	TRUE　　　　　：接続中 FALSE　　　　　：切断中
解説	USB デバイス状態を取得します。 接続中になると USB 通信が可能になります。

MyUsbFunctionSetSendData

データ送信関数

書式	BOOL MyUsbFunctionSetSendData(unsigned char *pData, unsigned long length)
引数	char * pData　　I　送信データバッファポインタ unsigned long length　　I　送信データサイズ(1～0x10000)
戻り値	TRUE　：正常終了 FALSE　：異常終了 ・ 送信データバッファが NULL ・ 送信データ長が 0 又は 0x10001 以上 ・ 接続が切れている場合 ・ 送信 FIFO がいっぱい
解説	指定されたデータをデータサイズ分、「まい USB」にてホストへ送信します。ホスト側ではMyUsbHostRecvData関数でデータを読み出してください。 MyUsbFunctionGetSendStatus関数をコールし“SENDING”でないことを確認してからデータバッファの内容を更新し、本関数を実行してください。 一回のデータ送信サイズは 0x10000Byte までですので、0x10000Byte を超えるデータを送信する場合には、複数回コールしてください。

MyUsbFunctionGetSendStatus

データ送信状態取得関数

書式	unsigned char MyUsbFunctionGetSendStatus(void)
引数	なし
戻り値	SEND_FINISH　(0)　：初期状態/送信完了 SENDING　(1)　：送信中 SEND_FAIL　(2)　：送信失敗
解説	データ送信状態を返します。 データ送信時には、本関数をコールし“SENDING”ではないことを確認してからMyUsbFunctionSetSendData関数コールしてください。 データ送信中に USB ケーブル切断されたときに、SEND_FAIL が返ります。 再接続時にも SEND_FAIL のままとなります。

MyUsbFunctionGetRecvData

データ受信関数

書式	unsigned char MyUsbFunctionGetRecvData(unsigned char *pData)
引数	char * pData　　　　　　　0　　受信バッファポインタ※
戻り値	受信データサイズ 以下の場合には、「0」が返ります。 ・ 受信バッファポインタが「NULL」 ・ 接続が切れた場合 ・ 受信データがない場合
解説	ホストからMyUsbHostSendData関数によって送信されたデータを「まいUSB」にて受信FIFOに格納します。 本関数は、受信FIFOに入っているデータを読み出し、受信データバッファにデータを格納します。 ※本関数を呼び出すときには、アプリケーションで静的に512Byte以上を確保してから本関数をコールするか、MyUsbFunctionGetRecvDataLength関数をコールし、存在している受信データ長分のバッファを確保してから本関数をコールしてください。

＊ホスト側から送信したデータがマイコンのFIFOサイズより大きい場合は、MyUsbFunctionGetRecvDataLengthの戻り値が“0”になるまでMyUsbFunctionGetRecvDataを複数回呼び出してすべてのデータを取得して下さい。

MyUsbFunctionGetRecvDataLength

受信データ長取得関数

書式	unsigned char MyUsbFunctionGetRecvDataLength(void)
引数	なし
戻り値	受信したデータ長
解説	MyUsbFunctionGetRecvData関数を呼び出した時に得られる受信 FIFO 上の受信バッファポインタが示すデータの長さをバイト単位で返します。

MyUsbFunctionGetRecvStatus

データ受信状態取得関数

書式	unsigned char MyUsbFunctionGetRecvStatus (void)
引数	なし
戻り値	RECV_OK　(1)　：正常 RECV_OVERFLOW　(0)　：受信FIFOがオーバーフローしている （ホストからの送信データを取りこぼしている。）
解説	データ受信状態を返します。 本関数にて受信 FIFO がオーバーフローしていたかどうかを取得することができます。 アプリケーションでホストから再送してもらうかどうかを知る手段としてご使用になれます。

MyUsbFunctionClearRecvErrorStatus

データ受信状態クリア関数

書式	void MyUsbFunctionClearRecvErrorStatus (void)
引数	なし
戻り値	なし
解説	データ受信状態を正常に戻します。 本関数をコールするまで、データ受信状態が保持されます。

MyUsbFunctionGetRecvFifoCouter

受信FIFO数取得関数

書式	unsigned long MyUsbFunctionGetRecvFifoCouter(void)
引数	なし
戻り値	受信データ数（0～129）
解説	ライブラリ内で管理している受信FIFO内に格納している受信データの個数を返します。

MyUsbFunctionSetPriority
USB 割り込み優先度設定関数

書式	void MyUsbFunctionSetPriority(unsigned short priority)		
引数	unsigned short priority	I	USB 割り込み優先度 （レベル：0～15）
戻り値	なし		
解説	指定された USB 割り込み優先度を設定します。 15 以上の値を設定した場合には、「15」を設定します。 ※MyUsbFunctionInit関数をコールした場合には「15」が設定されますので、USB 割り込み優先度を変更する場合には、MyUsbFunctionInit関数をコールした後に本関数をコールしてください。		

usb_function_interrupt
USB 割り込み処理関数

書式	void usb_function_interrupt(void)
引数	なし
戻り値	なし
解説	USB 割り込み処理を行います。 ・ USB ホストへの接続検出 ・ USB コネクタからの切断検出 ・ 認証処理 ・ 送信 FIFO に格納されているデータをホストへ送信する処理 ・ ホストからデータを受信し、受信 FIFO の格納する処理 ※本関数は、API 関数ではございませんので、アプリケーションからの関数コールは行わないでください。

9. ターゲット基板別制約

〇SH7262/7264 の場合

特になし。

10. 付録

CD に付属していますサンプルアプリケーションについて説明します。
尚、サンプルアプリケーションは、北斗電子製の HSB7264 で動作するアプリケーションです。

サンプルアプリケーションの実行環境を図 8に示します。

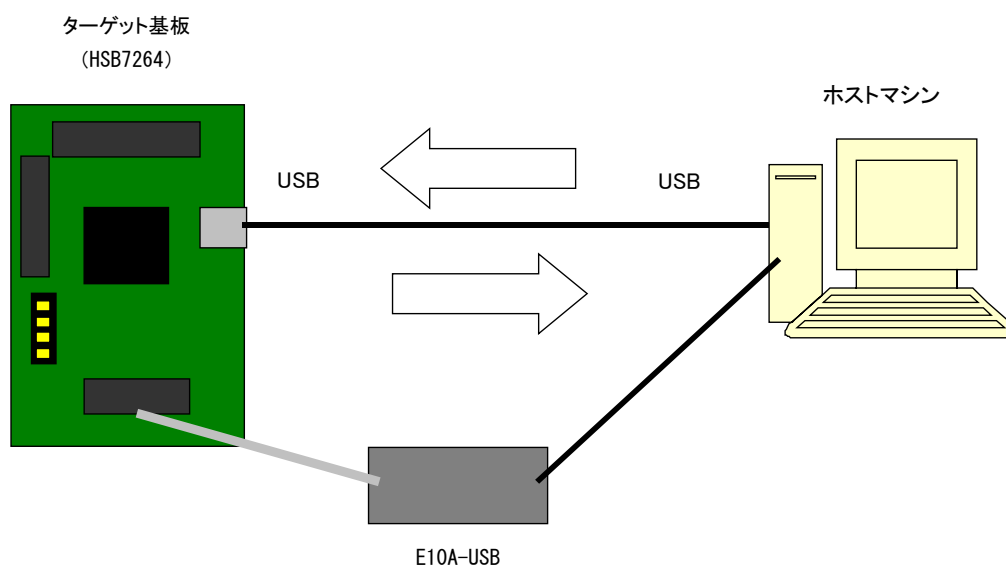



図 8　サンプルアプリケーションの実行環境

10.1.サンプルアプリケーションについて

サンプルアプリケーションは､HSB7264 とホストマシンで簡単なデータ転送を行うプログラムです。

■実行方法

サンプルアプリケーションを実行するための手順を以下に示します。

（1）CD に入っています「sample /Host」をホストマシンにコピーしてください。
（2）CD に入っています「sample /Function」をホストマシン（以下、デバッグ用マシン）にコピーしてください。
（3）デバッグ用マシンとターゲット基板を E10A-USB にて接続してください。
（4）ホストマシンとターゲット基板を USB ケーブルで接続し、ターゲット基板の電源を投入してください。
（5）デバッグ用マシンにて HEW を起動し、ターゲットファイルをダウンロードしてから実行してください。
（6）ホストマシンにてドライバのインストールが促されますので、①でコピーした「myusb_XXXX .inf」と「myusb_XXXX.sys」をイントールしてください。
（7）インストール完了後、①でコピーした「sample.exe」を起動してください。
（8）「sample.exe」から「接続」ボタンを押下してください。
（9）HEW から「SndFlag」を「0」から「1」に変更しますと、「「1」を受信しました。」のメッセージを表示されます。

「sample.exe」から「「0」送信」、「「1」送信」、「「2」送信」、「「3」送信」のいずれかを選択し、「実行」ボタンを押下すると HSB7264 のシリアルから対応した文字が出力されます。

10.2. 64bit ドライバ版の注意事項

まい USB(64bit 版)はドライバソフト(myusb_XXXX.sys)が 64bit 版のドライバとなります。
Windows アプリケーション作成時に用いる DLL(myusb_XXXX.dll)は 32bit 版となりますので、64bit 環境においてもプログラムのビルドは 32bit アプリケーションとして行ってください。

11. 著作権・免責について

本書の内容は予告無く変更する場合があります。本書は著作権により保護されています。
株式会社北斗電子(以下当社)の文書による事前の許諾無しに、本書を複写または複製、転載する事は禁じられています。

当社は本書の内容について万全を期して作成し、正確と確信しておりますが、当社による本書に関する保証は一切なく、特定の目的の市販性、正当性、適合性に関する黙示の保証に対する責任を否認します。本書の内容は予告無しに変更する事があります。本書の中に誤りがある場合でも、当社はいかなる責任も負いません。

本書の内容に関するお問い合わせはご容赦下さい。

本ソフトウェアをご利用になる際、別紙「ソフトウェア使用権許諾契約書」を熟読ください。
本ソフトウェアの価格にはサポート料は含まれておりません。また、現地調査等が必要な場合はサポート料＋交通費や宿泊費がかかります。

最新情報については弊社ホームページをご活用下さい。
URL:http://www.hokutodenshi.co.jp

まい USB　取扱説明書

発行　株式会社北斗電子
〒060-0042 札幌市中央区大通西 16 丁目 3 番地 7
TEL：011-640-8800
FAX：011-640-8801
e-mail：support@hokutodenshi.co.jp（サポート用）
order@hokutodenshi.co.jp（ご注文用）
URL:http://www.hokutodenshi.co.jp